



The Host



Jennifer Rexford

Fall 2010 (TTh 1:30-2:50 in COS 302)

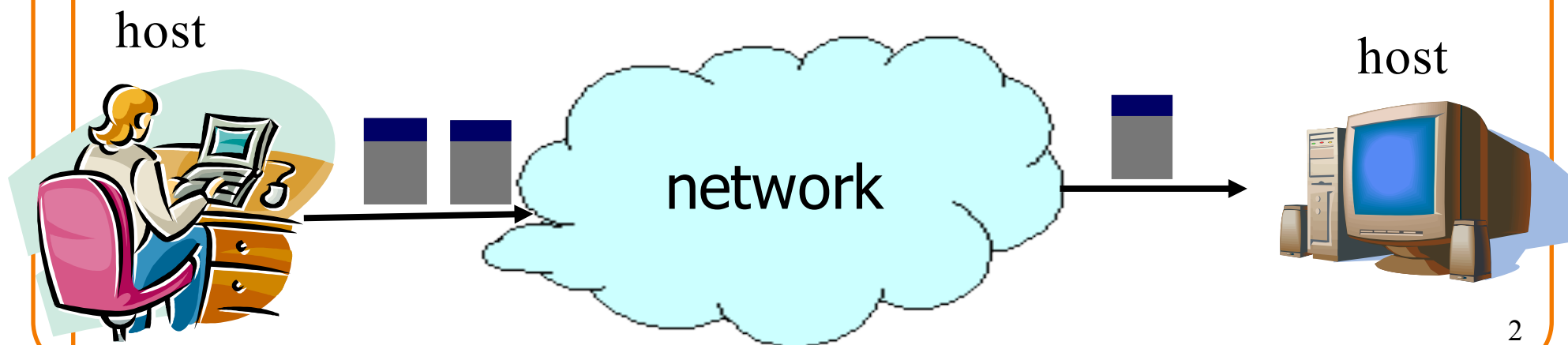


COS 561: Advanced Computer Networks

<http://www.cs.princeton.edu/courses/archive/fall10/cos561/>

Host-Network Division of Labor

- Network
 - Best-effort packet delivery
 - Between two (or more) end-point addresses
- Hosts
 - Everything else





The Role of the End Host

- Network discovery and bootstrapping
 - How does the host join the network?
 - How does the host get an address?
- Interface to networked applications
 - What interface to higher-level applications?
 - How does the host realize that abstraction?
- Distributed resource sharing
 - What roles does the host play in network resource allocation decisions?



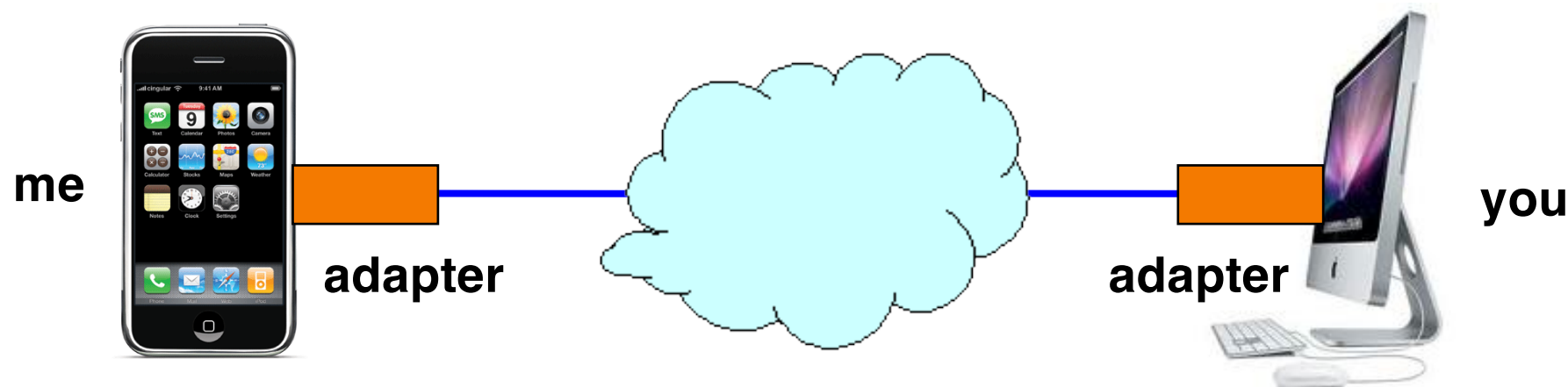
Network Discovery and Bootstrapping



Three Kinds of Identifiers

	Host Name	IP Address	MAC Address
Example	www.cs.princeton.edu	128.112.7.156	00-15-C5-49-04-A9
Size	Hierarchical, human readable, variable length	Hierarchical, machine readable, 32 bits (in IPv4)	Flat, machine readable, 48 bits
Read by	Humans, hosts	IP routers	Switches in LAN
Allocation, top-level	Domain, assigned by registrar (e.g., for .edu)	Variable-length prefixes, assigned by ICANN, RIR, or ISP	Fixed-sized blocks, assigned by IEEE to vendors (e.g., Dell)
Allocation, low-level	Host name, local administrator	Interface, by admin or DHCP	Interface, by vendor

Learning a Host's Address



- Who am I?
 - Hard-wired: MAC address
 - Static configuration: IP interface configuration
 - Dynamically learned: IP address configured by DHCP
- Who are you?
 - Hard-wired: IP address in a URL, or in the code
 - Dynamically looked up: ARP or DNS



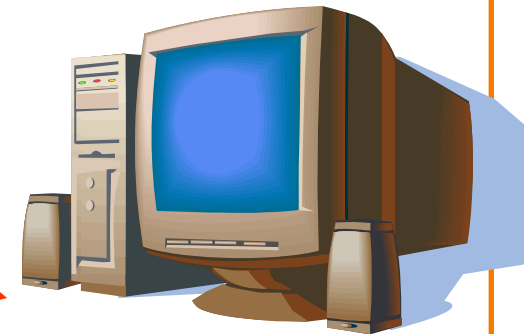
Mapping Between Identifiers

- **Dynamic Host Configuration Protocol (DHCP)**
 - Given a MAC address, assign a unique IP address
 - ... and tell host other stuff about the Local Area Network
 - To automate the boot-strapping process
- **Address Resolution Protocol (ARP)**
 - Given an IP address, provide the MAC address
 - To enable communication within the Local Area Network
- **Domain Name System (DNS)**
 - Given a host name, provide the IP address
 - Given an IP address, provide the host name

Dynamic Host Configuration Protocol



arriving
client



DHCP server

DHCP discover
(broadcast)

DHCP offer

DHCP request
(broadcast)

DHCP ACK

Host learns
IP address,
Subnet mask,
Gateway address,
DNS server(s),
and a lease time.

Address Resolution Protocol (ARP)

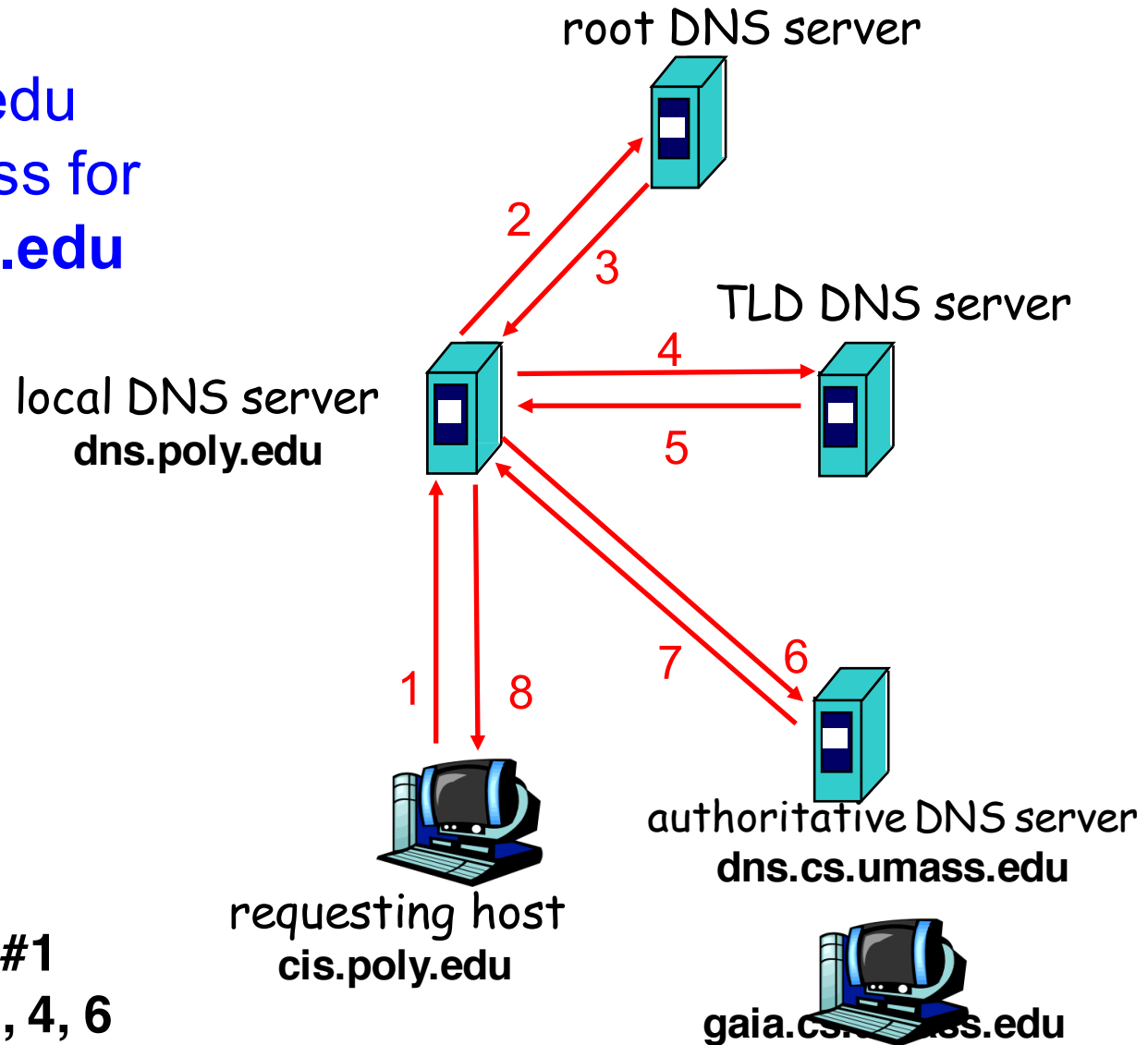


- Every host maintains an ARP table
 - (IP address, MAC address) pair
- Consult the table when sending a packet
 - Map destination IP address to destination MAC address
 - Encapsulate and transmit the data packet
- But, what if the IP address is not in the table?
 - Sender broadcasts: “Who has IP address 1.2.3.156?”
 - Receiver responds: “MAC address 58-23-D7-FA-20-B0”
 - Sender caches the result in its ARP table

Domain Name System



Host at cis.poly.edu
wants IP address for
gaia.cs.umass.edu



Recursive query: #1
Iterative queries: #2, 4, 6



Questions

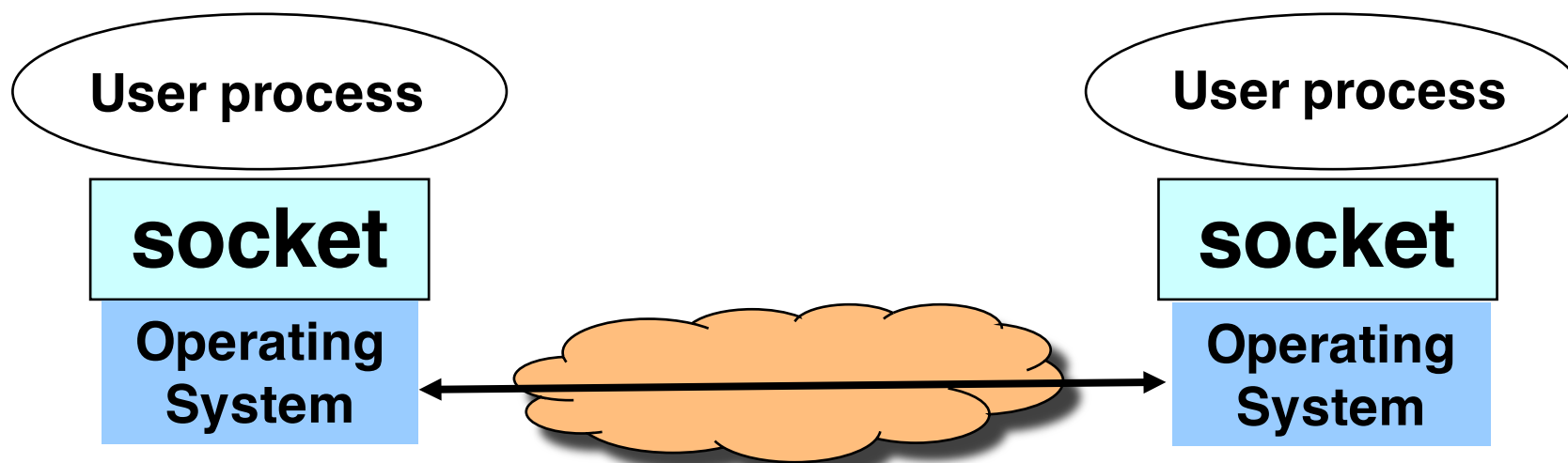
- Should addresses correspond to the interface (point of attachment) or to the host?
- Why do we have all three identifiers? Do we need all three?
- What should be done to prevent spoofing of addresses?



Interface to Applications

Socket Abstraction

- Best-effort packet delivery is a clumsy abstraction
 - Applications typically want higher-level abstractions
 - Messages, uncorrupted data, reliable in-order delivery

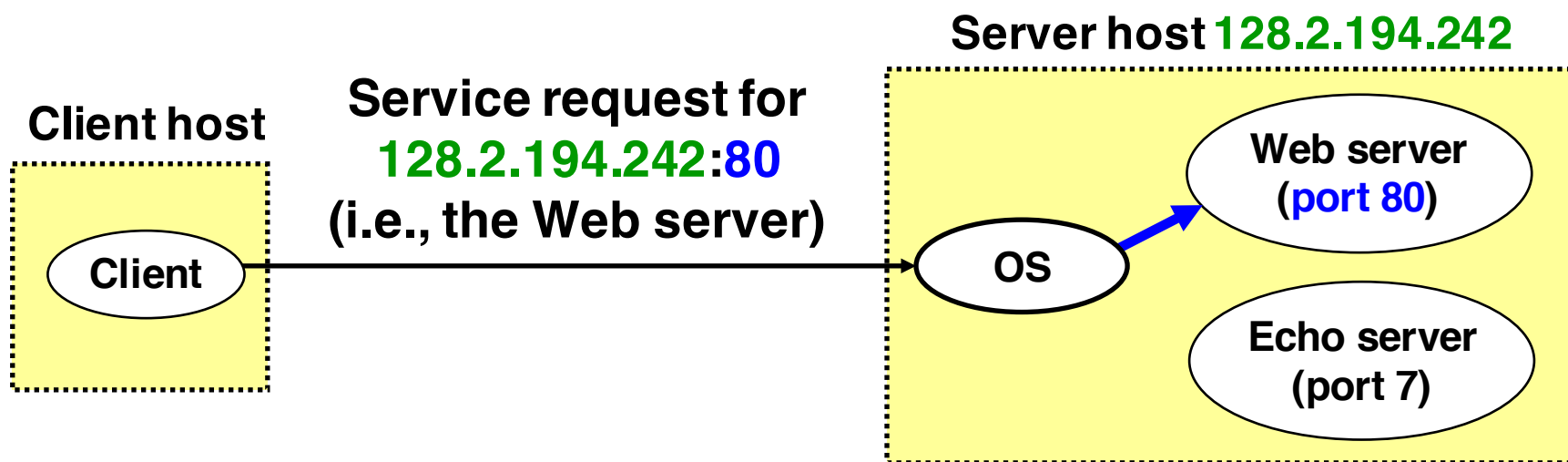


- Applications communicate using “sockets”
 - Stream socket: reliable stream of bytes (like a file)
 - Message socket: unreliable message delivery

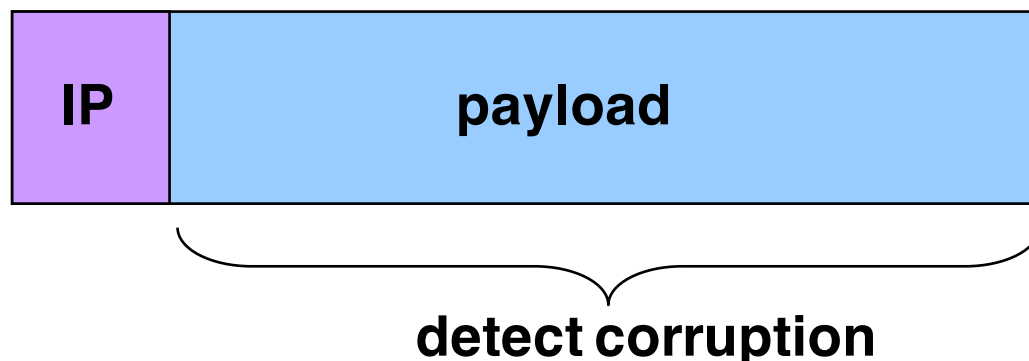


Two Basic Transport Features

- **Demultiplexing: port numbers**



- **Error detection: checksums**





Two Main Transport Layers

- User Datagram Protocol (UDP)
 - Just provides demultiplexing and error detection
 - Header fields: port numbers, checksum, and length
 - Low overhead, good for query/response and multimedia
- Transmission Control Protocol (TCP)
 - Adds support for a “stream of bytes” abstraction
 - Retransmitting lost or corrupted data
 - Putting out-of-order data back in order
 - Preventing overflow of the receiver buffer
 - Adapting the sending rate to alleviate congestion
 - Higher overhead, good for most stateful applications



Questions

- Is a socket between two IP addresses the right abstraction?
 - Mobile hosts?
 - Replicated services?
- What does the network know about the traffic?
 - Inferring the application from the port numbers?
- Is end-to-end error detection and correction the right model?
 - High loss environments?
 - Expense of retransmitting over the entire path?

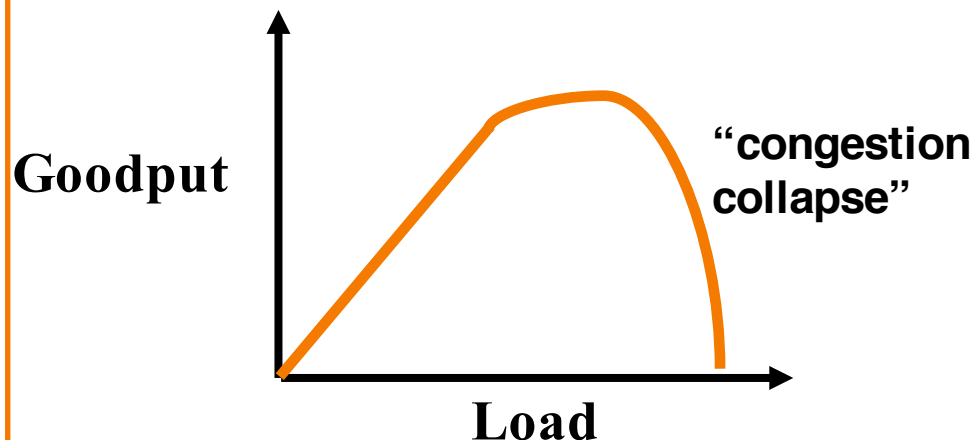


Distributed Resource Sharing

Resource Allocation Challenges



- Best-effort network easily becomes overloaded
 - No mechanism to “block” excess calls
 - Instead excess packets are simply dropped
- Examples
 - Shared Ethernet medium: frame collisions
 - Ethernet switches and IP routers: full packet buffers
- Quickly leads to congestion collapse



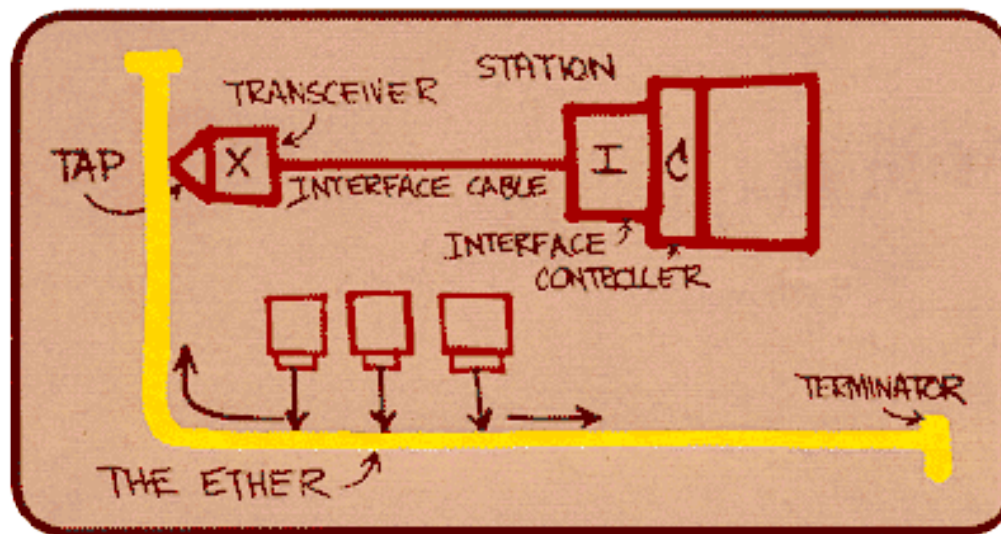
Increase in load that results in a *decrease* in useful work done.

End Hosts Adjusting to Congestion



- End hosts adapt their sending rates
 - In response to network conditions
- Learning that the network is congested
 - Shared Ethernet: carrier sense multiple access
 - Seeing your own frame collide with others
 - IP network: observing your end-to-end performance
 - Packet delay or loss over the end-to-end path
- Adapting to congestion
 - Slowing down the sending rate, for the greater good
 - But, host doesn't know how bad things might be...

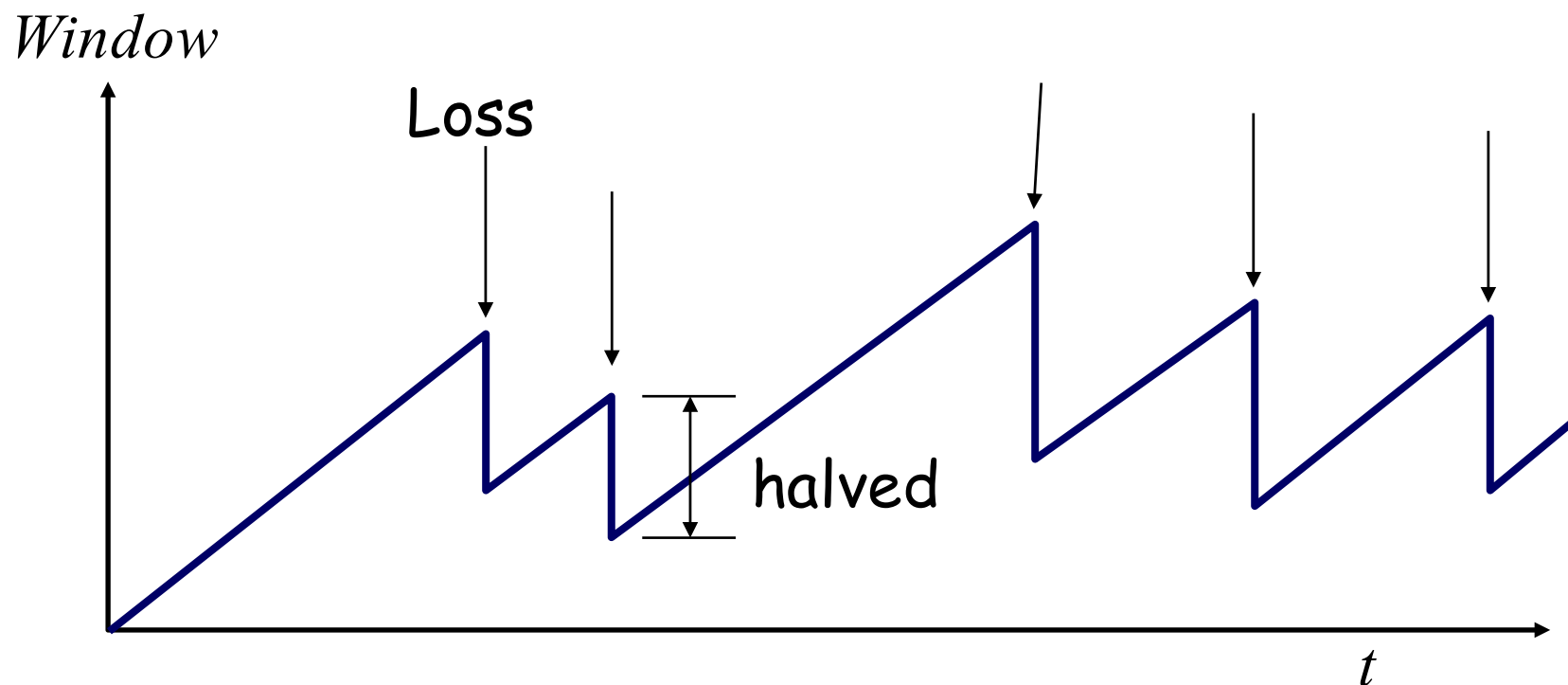
Ethernet Back-off Mechanism



- Carrier sense: wait for link to be idle
 - If idle, start sending; if not, wait until idle
- Collision detection: listen while transmitting
 - If collision: abort transmission, and send jam signal
- Exponential back-off: wait before retransmitting
 - Wait random time, exponentially larger on each retry

TCP Congestion Control

- Additive increase, multiplicative decrease
 - On packet loss, divide congestion window in half
 - On success for last window, increase window linearly



Other mechanisms: slow start, fast retransmit vs. timeout loss, etc. ²¹



Questions

- What role should the network play in resource allocation?
 - Explicit feedback to the end hosts?
 - Enforcing an explicit rate allocation?
- What is a good definition of fairness?
- What about hosts who cheat to hog resources?
 - How to detect cheating? How to prevent/punish?
- What about wireless networks?
 - Difficulty of detecting collisions (due to fading)
 - Loss caused by interference, not just congestion



“A Protocol for Packet Network Intercommunication”

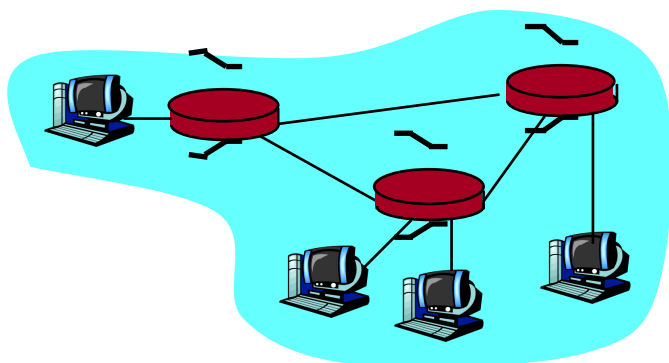
(IEEE Trans. on Communications, May 1974)

Vint Cerf and Bob Kahn

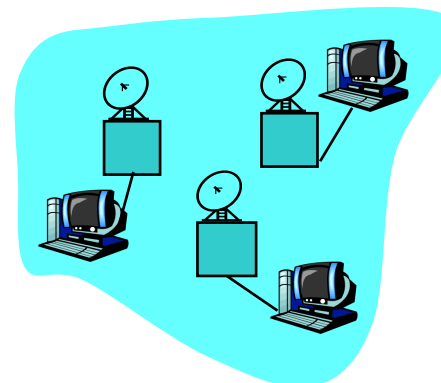
Written when Vint Cerf was an assistant professor at Stanford, and Bob Kahn was working at ARPA.

Life in the 1970s...

- Multiple unconnected networks
 - ARPAnet, data-over-cable, packet satellite (Aloha), packet radio, ...
- Heterogeneous designs
 - Addressing, max packet size, handling of lost/corrupted data, fault detection, routing, ...



ARPAnet



satellite net



Handling Heterogeneity

- Where to handle heterogeneity?
 - Application process? End hosts? Packet switches?
- Compatible process and host conventions
 - Obviate the need to support all combinations
- Retain the unique features of each network
 - Avoid changing the local network components
- Introduce the notion of a gateway

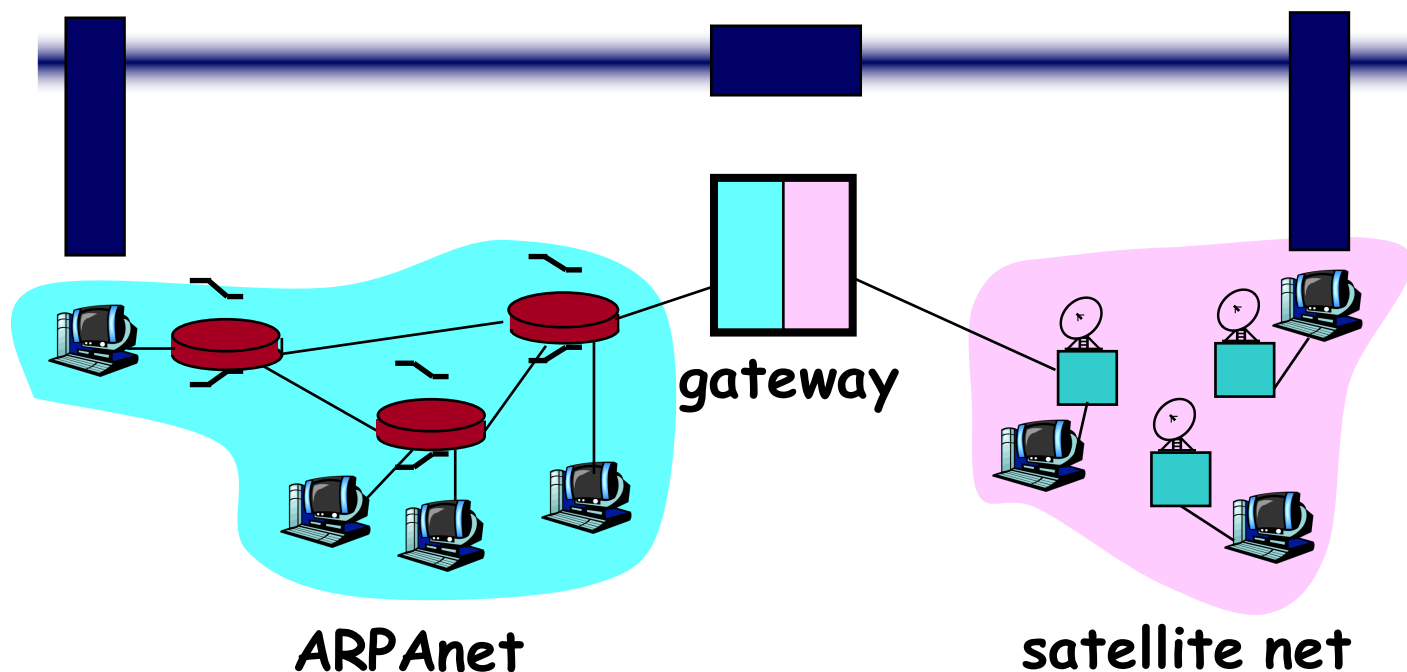
Internetwork Layer and Gateways

Internetwork Layer

- Internetwork appears as a single, uniform entity
- Despite the heterogeneity of the local networks
- Network of networks

Gateway

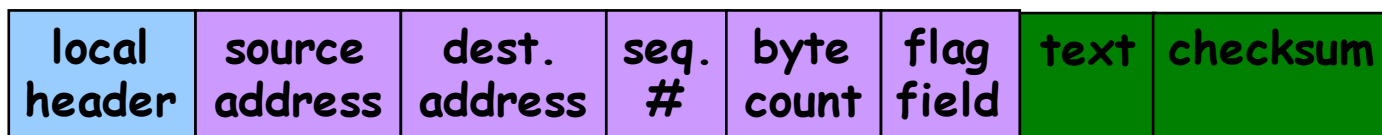
- “Embed internetwork packets in local packet format or extract them”
- Route (at internetwork level) to next gateway





Internetwork Packet Format

internetwork header



- Internetwork header in standard format
 - Interpreted by the gateways and end hosts
- Source and destination addresses
 - Uniformly and uniquely identify every host
- Ensure proper sequencing of the data
 - Include a sequence number and byte count
- Enable detection of corrupted text
 - Checksum for an end-to-end check on the text



Process-Level Communication

- Enable pairs of processes to communicate
 - Full duplex
 - Unbounded but finite-length messages
 - E.g., keystrokes or a file
- Key ideas
 - Port numbers to (de)multiplex packets
 - Breaking messages into segments
 - Sequence numbers and reassembly
 - Retransmission and duplicate detection
 - Window-based flow control

Discussion



- What did they get right?
 - Which ideas were key to the Internet's success?
 - Which decisions still seem right today?
- What did they miss?
 - Which ideas had to be added later?
 - Which decisions seem wrong in hindsight?
- What would you do in a clean-slate design?
 - If your goal wasn't to support communication between disparate packet-switched networks
 - Would you do anything differently?



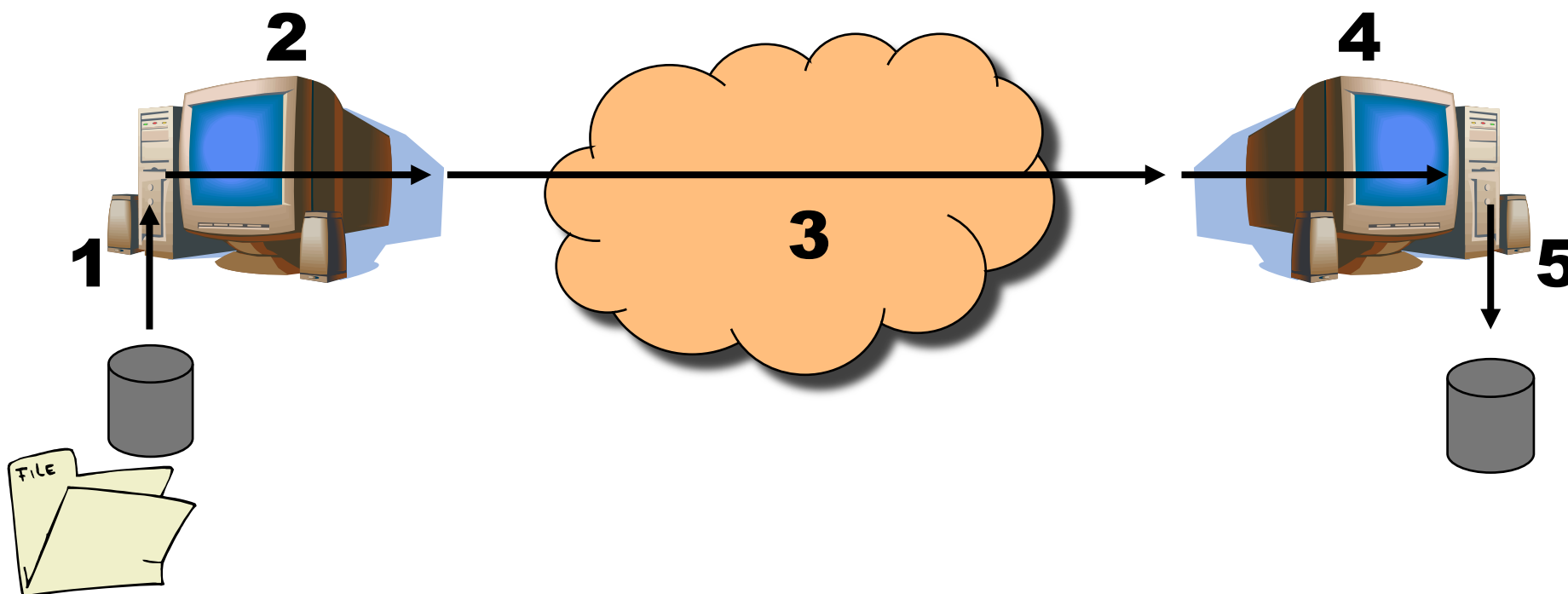
“End-to-End Arguments in System Design”

(ACM Trans. on Computer Systems, November 1984)

J. Saltzer, D. Reed, and D. Clark

End-to-End Argument

- Operations should occur only at the end points
- ... unless needed for performance optimization



Many things can go wrong: disk errors, software errors, hardware errors, communication errors, ...



Trade-Offs

- Put functionality at each hop
 - All applications pay the price
 - End systems *still* need to check for errors
- Place functionality only at the ends
 - Slower error detection
 - End-to-end retransmission wastes bandwidth
- Compromise solution?
 - Reliable end-to-end transport protocol (TCP)
 - Plus file checksums to detect file-system errors

Discussion



- When should the network support a function anyway?
 - E.g., link-layer retransmission in wireless networks?
- Who's interests are served by the e2e argument?
- How does a network operator influence the network without violating the e2e argument?
- Does the design of IP and TCP make it *hard* to violate the e2e argument?
 - E.g., middlebox functionality like NATs, firewalls, proxies
- Should the e2e argument apply to routing?